New Technical Notes

Macintosh

Developer Support

Movie Toolbox Q&As

QuickTime

Revised by: Developer Support Center June 1993
Written by: Developer Support Center October 1990

This Technical Note contains a collection of Q&As relating to a specific topic—questions you've sent the Developer Support Center (DSC) along with answers from the DSC engineers. While DSC engineers have checked the Q&A content for accuracy, the Q&A Technical Notes don't have the editing and organization of other Technical Notes. The Q&A function is to get new technical information and updates to you quickly, saving the polish for when the information migrates into reference manuals.

Q&As are now included with Technical Notes to make access to technical updates easier for you. If you have comments or suggestions about Q&A content or distribution, please let us know by sending an AppleLink to DEVFEEDBACK. Apple Partners may send technical questions about Q&A content to DEVSUPPORT for resolution.

New Q&As in this Technical Note:
QuickTime rotation and skew aren't implemented
QuickTime 1.6 fixes GetMoviePosterPict bug

QuickTime rotation and skew aren't implemented

Date Written: 1/22/93
Last reviewed: 4/1/93

We can't apply the rotation and skew effects to a QuickTime 1.5 movie. We've created an identity matrix, applied RotateMatrix to the matrix, set the matrix to the movie using SetMovieMatrix, and played the movie. The movie didn't rotate but the movieRect rotated and the movie scaled to the movieRect. Is there anything wrong with what we're doing?

—

Rotation and skew don't work because nonlinear transformations haven't been implemented into QuickTime 1.0 or 1.5. The only transformations that work are scaling and offset. Rotation and skew are important future directions. However, if you use QuickDraw GX, you can accomplish the rotation and skewing by playing a movie to an off-screen and then using GX to display the off-screen rotated or skewed.

QuickTime 1.6 fixes GetMoviePosterPict bug

Date Written: 1/11/93
Last reviewed: 4/1/93

I’m opening a movie document and using NewMovieFromFile to get a movie. When I try to get the poster for the movie I get back a paramErr (-50) from the GetMoviesError call and a NIL picHandle from GetMoviePosterPict. If I use QuickTme 1.0 everything works fine but QuickTime 1.5 gives me the paramErr. Am I doing something wrong or is there a bug?

—

You’re hitting your head against a QuickTime 1.5 bug that (as far as we can tell) has no reasonable workaround. It has to do with asking for the PICT from a movie at a time (movie time) that falls in between differentiated frames of the same size. The problem has been identified and fixed in QuickTime 1.6.

Check whether 'vdig' supports off-screen digitizing
Date Written: 1/7/93
Last reviewed: 3/1/93

When we try to digitize frames (grabbed using QuickTime) into an off-screen pixmap, our VDGrabOneFrame call crashes. How would you suggest I digitize into an off-screen pixmap?

—

You need to check whether or not the 'vdig' resource supports digitizing to off-screens by calling the VDPreflightDestination call! If the PreflightDestination call with an off-screen destination fails, then you need to digitize to a window on the digitizing device, and then copy (using CopyBits or your own algorithm for speed) the image from the window to your off-screen. Some 'vdig' resources don’t support digitizing to off-screen directly because their hardware does the digitization asynchronously. So, your application will have to do some extra work in some cases. (You should always preflight your destination before setting it with SetPlayThruDestination.)

Techniques for controller to go to beginning or end
Date Written: 11/24/92
Last reviewed: 3/1/93

The controller constants have been left out of the latest Movies.h header for Goto beginning and ending commands. Only mcGotoTime is left. Is it OK to use these constants or is there a newly defined and accepted way to do this now?

—

Scanning the interfaces, we can’t locate a time when the controller had actions to go to the beginning or end. In any case there are two ways of accomplishing the same objective: The first is to use the MCActionGotoTime and pass the zero to go to the start of the movie and the result from GetMovieDuration to go to the end of the movie. The code to do this gets a

little bit cumbersome since you have to pass a TimeRecord. This may be why our favorite is the second method.

Second, to go to the start, call GoToBeginningOfMovie followed by MCMovieChanged. And to go to the end, call GoToEndOfMovie also followed by MCMovieChanged. The following routine shows how to accomplish what you want:

```
void DoGotoEnds(short direction)
```

```
{
WindowPtr window;
DocRecHandle wHndl;

 if (window = FrontWindow()) { /* don't bother if no movies to play */
   if (IsAppWindow(window) && (wHndl = (DocRecHandle)GetWRefCon(window)) ) {
     switch (direction) {
       case gotoBeginning:
         GoToBeginningOfMovie( (*wHndl)->wMovie);
         break;
       case gotoEnd:
         GoToEndOfMovie( (*wHndl)->wMovie );
         break;
       default:
         return; /* do nothing if came here on error */
         break;
     }
   MCMovieChanged ((*wHndl)->wPlayer, (*wHndl)->wMovie);
   }
 }
}
```

Doing this you let the Movie Toolbox take care of all the details and the results are the same. MCMovieChanged is needed in order to make the controller aware of the different state.

As a bonus, the following code shows (in a different scenario) how you can fill in the time record necessary for the MCActionGoto call in case that's the route you want to take:

```
/* selects the whole movie for editing */
void DoSelectAll(void)
{
WindowPtr window;
DocRecHandle wHndl;
ComponentResult err;
TimeRecord tRec;

 if (window = FrontWindow()) { /* don't bother if no movies to play */
   if (IsAppWindow(window) && (wHndl = (DocRecHandle)GetWRefCon(window)) ) {
     if ( MCIsEditingEnabled((*wHndl)->wPlayer)) {
       tRec.value.lo = tRec.value.hi = 0;
       tRec.base = 0;
       tRec.scale = GetMovieTimeScale((*wHndl)->wMovie);
       if ( err = MCDoAction((*wHndl)->wPlayer, mcActionSetSelectionBegin, &tRec)) {
         DebugStr("\pError trying mcActionSetSelectionBegin");
       }

       tRec.value.hi = 0;
       tRec.value.lo = GetMovieDuration((*wHndl)->wMovie);
       tRec.base = 0;
       tRec.scale = GetMovieTimeScale((*wHndl)->wMovie);
       if ( err = MCDoAction((*wHndl)->wPlayer, mcActionSetSelectionDuration, &tRec)) {
         DebugStr("\pError trying mcActionSetSelectionDuration");
       }
     }
   }
 }
}
```

Getting QuickTime playback frame rate on the fly
Date Written: 11/24/92
Last reviewed: 3/1/93

I need both to get and set frames/sec for my frames-per-second playback rate menu. GetMovieRate and SetMovie rate seem too transient to use since the movie has to be currently playing and the effect leaves once it stops. The PreferredRate calls look good but are only scalars. How do I get the real frames/second and how do I actually set a rate that the controller will honor when the Play button is pressed?

—

At the moment there are no facilities to ascertain the real frame rate while the movie is playing. The view from engineering is that with enough interest and good reasons for this to be a feature, it may get into a future version of QuickTime but at the moment it can't be done.

Setting the rate is even more remotely in the future. As you know, the movie toolbox drops frames as needed to keep pace with the passing of time. This is a self-regulating process and there's no way to tell the toolbox how many frames you want to see because it may not be able to deliver based on certain conditions.

Interleave QuickTime 1.5 movie sound and video data
Date Written: 11/20/92
Last reviewed: 3/1/93

Our product's sound/video synchronization is way off with QuickTime 1.5; it worked perfectly with QuickTime 1.0. The video can't keep up with the sound, especially on the full-screen movies. The movies are playing much slower with 1.5. Isn't QuickTime 1.5 supposed to make movies play faster?

—

Your movies probably aren't properly interleaved. When you add sound to a movie using SoundToMovie, the sound is added to the end of the video data. We recommend that sound and video be interleaved so that the hard drive doesn't have to spend extra time seeking between media that store video and media that store audio on a hard drive. The data handler prefers to be able to sequentially read through a movie file. This is especially important for slower Macintosh models that don't have the extra CPU and SCSI access speed to spare.

QuickTime can accommodate for some noninterleaved data by caching an entire sound track of a movie if small enough. However, the size of a cache is internal to QuickTime and can't be depended on. It's possible that different QuickTime versions could have different cache sizes since we've been recommending that video and sound movies be interleaved. The result could be that the extra disk-seeking time has caused sound and video to be out of sync for slower machines such as the Macintosh LC.

One way to check interleaving is to resave the movie using Movie Converter (or some other program that flattens movies) in a flattened format. Movie Converter uses QuickTime's FlattenMovie call to do this. The steps Movie Converter takes are: choose Save As, select Make movie self-contained, and save to a new movie file. This new movie should play back with correct video and sound sync.

You can actually see the problem if you examine the movie with MovieShop, a program that lets you deal with QuickTime movies at a movie data level. For example, if you select the Play information button that's in the window after you open a movie, the program will display a time graph showing you where the video and sound data are saved in the continuous data stream. If the movie is interleaved, the green (for video) and red (for sound) indicators are interleaved. If the movie isn't interleaved, the green indicators are clumped together in the beginning of the file, and the red indicators (for sound) are at the end.

The latest version of MovieShop and documentation is available on this issue's CD. MovieShop and information relating to this Q&A are discussed in the article "Making Better QuickTime Movies" in this issue.

Saving a movie's active selection
Date Written: 11/2/92
Last reviewed: 3/1/93

I have a movie with two video tracks: Track #1—Enabled, Duration=10; Track #2—Disabled, Duration=20. I set the movie's active segment to (0, 10) and saved the movie resource to a movie file. When I open the movie in MoviePlayer, the movie is played for the time value of 20, ignoring the active segment and there being one enabled track with a duration of 10. Short of creating a new movie from a selection of the above mentioned movie, is there any way to get MoviePlayer to do what I'd expect and not ignore the active segment and play past the enabled track's duration?

—

The active selection isn't saved along with a movie. Therefore, no application will be able to restore and play back the active segment. You'd have to create a new movie from the selection in order to get MoviePlayer or any other application to play that selection only. If users will be using your application to play back the movie, you could store information regarding the active segment in a user data atom inside the movie. You could then have your application load the user data atom if it exists inside a movie when it's opened, to restore the selection. That would be the only way to save the active movie selection.

How to get movie frame time parameter for GetMoviePict
Date Written: 9/9/92
Last reviewed: 3/1/93

How do I find the correct time values to pass to GetMoviePict, to get all the sequential frames of a QuickTime movie?

—

The best way to find the correct time to pass to get movie frames is to call the GetMovieNextInterestingTime routine repeatedly. Note that the first time you call

GetMovieNextInterestingTime its flags parameter should have a value of nextTimeMediaSample+nextTimeEdgeOK to get the first frame. For subsequent calls the value of flags should be nextTimeMediaSample. Also, the whichMediaTypes parameter should include only tracks with visual information, 'vide' and 'text'. Check the Movie Toolbox chapter of the QuickTime documentation for details about the GetMovieNextInterestingTime call. For a code example, see the revised SimpleInMovie on your developer CD. The routine to look at is called DoGetMoviePicts in the file SimpleInPicts.c.

Determining QuickDraw video media pixel depth
Date Written: 8/18/92
Last reviewed: 1/11/93

How do I get the pixel depth of the QuickTime video media for a given track?

—

To find the video media pixel depth, you'll need to retrieve the media's image description handle. You can use GetMediaSampleDescription to get it, but this routine needs both the video media and the track's index number. It's not obvious, but a media's type is identified by its media handler's type. Thus, you can walk through a movie's tracks by using its indexes until you find video media, at which point you have both the track index and video media.

The following sample code does the trick:

```
Media GetFirstVideoMedia(Movie coolMovie, long *trackIndex)
{
 Track    coolTrack = nil;
 Media    coolMedia = nil;
 long    numTracks;
 OSType  mediaType;
 numTracks = GetMovieTrackCount(coolMovie);
 for (*trackIndex=1; *trackIndex<=numTracks; (*trackIndex)++) {
   coolTrack = GetMovieIndTrack(coolMovie, *trackIndex);
   if (coolTrack) coolMedia = GetTrackMedia(coolTrack);
   if (coolMedia) GetMediaHandlerDescription(coolMedia,
     &mediaType, nil, nil);
   if (mediaType = VideoMediaType) return coolMedia;
 }
 *trackIndex = 0;  // trackIndex can't be 0
 return nil;      // went through all tracks and no video
}

short GetFirstVideoTrackPixelDepth(Movie coolMovie)
{
 SampleDescriptionHandle imageDescH =
   (SampleDescriptionHandle)NewHandle(sizeof(Handle));
 long    trackIndex = 0;
 Media    coolMedia = nil;
 coolMedia = GetFirstVideoMedia(coolMovie, &trackIndex);
 if (!trackIndex || !coolMedia) return -1;  // we need both
 GetMediaSampleDescription(coolMedia, trackIndex, imageDescH);
 return (*(ImageDescriptionHandle)imageDescH)->depth;
}
```

QuickTime 1.5 fixes GetMoviePict grayscale images -157 error
Date Written: 8/27/92
Last reviewed: 1/11/93

Trying to read any single frame (except the first) from a grayscale (4, 16, or 256 grays), QuickTime movie gives a -157 error (invalid pixel depth). I do "framehandle = GetMoviePict(damovie,daunitnum)" and test if it's 0. If so, then I do "io = GetMoviesError()" and it gives me the -157 error. GetMoviePict is working fine for all color

and black/white

movies. It just doesn't work on "Animation" grayscale and "Video" grayscale movies. Is this a bug?

—

With QuickTime 1.0, grayscale images cause a -157 error with GetMoviePict; some grayscale images may work if the color tables are custom made (ImageDescription structure clutID is 0). However, QuickTime 1.5 fixes this problem.

Determining whether a movie is set to loop or not
Date Written: 8/11/92
Last reviewed: 3/11/93

How does Simple Player determine whether a movie is set to loop or not? Movie files that are set to loop seem to have a string of 'LOOP' at the end of the 'moov' resource. Does Simple Player check 'LOOP'?

—

Simple Player identifies whether movies are set to loop by looking within the user data atoms for the 'LOOP' atom, as you've noticed. It's a 4-byte Boolean in which a value of 1 means standard looping and a value of 0 means palindrome looping. Your applications should add the user data 'LOOP' atom to the end of the movie when a user chooses to loop. We recommend this method as a standard mechanism for determining the looping status of a movie. If the 'LOOP' atom doesn't exist, there's no looping. The calls you need to access this information are GetMovieUserData, GetUserData, AddUserData, and RemoveUserData, as defined in the Movie Toolbox chapter of the QuickTime documentation. For more information see the Macintosh Technical Note "Movies 'LOOP' Atom and Friends."

Passing a refCon to the movie controller action filter proc
Date written: 7/22/92
Last reviewed: 1/11/93

Is there a way that the action filter proc of a QuickTime movie controller component can have a user reference field so that I can know which movie-"object" the movie controller is referring to? There are local variables associated with a particular movie that I would like to access from the action filter proc; currently there's no way to reference back to the variables in my program except through globals.

—

This was a difficult task under QuickTime 1.0, requiring you to stuff some sort of pointer in the movie user data fields. The good news is that at developer requests, QuickTime 1.5 includes a new call in the Movie Controller suit allowing you to pass and receive a long value when you set up your filter proc as follows:

```
pascal ComponentResult MCSetActionFilterWithRefCon (MovieController mc,
```

```
MCActionFilterWithRefCon blob, long refCon)  = {0x2F3C,0x8,0x2D,0x7000,0xA82A};
```

The proc is of the form:

```
pascal Boolean userPlayerFilter(MovieController mc, short action, void *params,
long refCon);
```

The proc returns TRUE if it handles the action, FALSE if not. The “action” parameter identifies the action to be executed; “params” is the set of potential parameters that go with the action; and “refCon” is any long value passed to MCSetActionFilterWithRefCon.

Problem with disabling a movie video track

Date Written: 5/29/92
Last reviewed: 1/11/93

When I disable a track in a movie, another random track becomes disabled as well. Is this a QuickTime bug?

—

This was a QuickTime 1.0 bug, but it’s fixed with QuickTime 1.5. With QuickTime 1.5 it’s no longer necessary to work around the bug, since disable now works.

'pnot' resource format for QuickTime-like preview in dialog box

Date Written: 3/6/92
Last reviewed: 1/11/93

I would like to implement the preview/thumbnail feature in the Standard File dialog, just like the extension included with QuickTime. Is that code available separate from QuickTime? If not, could I at least get information on how the preview is created?

—

To implement your own preview/thumbnail feature, simply duplicate the Standard File dialog, add the necessary 'DITL' resources, and install a custom filter procedure for handling preview commands. On the System 7 CD there’s an example, StdFileSample, that shows exactly how to create a custom file dialog. The Macintosh Technical Note “Customizing Standard File” describes how to do this as well. For generating and displaying the preview, you can use the following PreviewResourceRecord, found at the end of the ImageCompression.h file:

```
struct PreviewResourceRecord {
   unsigned long        modDate;
   short                version;
   OSType               resType;
   short                resID;
};

typedef struct PreviewResourceRecord PreviewResourceRecord;
typedef PreviewResourceRecord *PreviewResourcePtr, **PreviewResource;
```

This is the format for the 'pnot' resource, which defines the preview for the movie file, usually pointing to a 'PICT' resource. It’s all you need to generate QuickTime-compatible preview without using QuickTime.

QuickTime alias and FSSpec system services under System 6

Date Written: 3/3/92
Last reviewed: 1/11/93

I recall reading that QuickTime includes an implementation of the Alias Manager for System 6, but I haven't found any precise description of what's included. Is it a bare minimum to support QuickTime? Or is the full Alias Manager there? Also, is there any way I can use the FSSpec interface to the File Manager, or must I revert to the System 6 interface?

—

When QuickTime is installed, most of the Alias Manager is available in System 6, with these exceptions:

• NewAlias will accept a fromFile parameter, but it never creates a relative alias.
• NewAliasMinimalFromFullPath and ResolveAliasFile aren't available.
• ResolveAlias and UpdateAlias will accept a fromFile parameter, but ignore it.
• MatchAlias may be called, but the kARMMultVols, kARMSearchMore, and kARMSearchRelFirst flags aren't available. If you pass them in, they'll be ignored. Furthermore, if you pass in a matchProc, it will never be called.
• The System 6 Alias Manager won't mount network volumes.

To summarize, in System 6 the Alias Manager doesn't handle relative aliases, multiple volume searches, "searchMore" searches, and network volume mounting. On the bright side, nearly all calls are present. Aliases created in System 6 are compatible with System 7 aliases, and aliases created in System 7 will work in System 6.

Unfortunately, QuickTime doesn't currently install an Alias Manager Gestalt selector, since it's only a partial implementation. You can check for the Alias Manager using Gestalt and, if it isn't present, look for QuickTime (using Gestalt); if QuickTime is present, assume you have an Alias Manager, subject to the limitations listed above.

QuickTime also makes extensive use of the FSSpec data structure introduced in the System 7 File Manager. Nearly all the FSSpec calls are available in System 6 when QuickTime is installed. The following calls are available in System 6, and should behave as documented for System 7: FSMakeFSSpec, FSpOpenDF, FSpOpenRF, FSpCreate, FSpDirCreate, FSpDelete, FSpGetFInfo, FSpSetFInfo, FSpSetFLock, FSpRstFLock, FSpRename, FSpCatMove, FSpOpenResFile, FSpCreateResFile, and FSpGetCatInfo. FSpExchangeFiles isn't available when using the QuickTime System 6 version of the FSSpec calls.

Again, the Gestalt selector for the FSSpec calls isn't installed when QuickTime is there. This means that the gestaltFSAttr Gestalt selector may not be present, and gestaltHasFSSpecCalls may not be set, even if gestaltFSAttr is present.

Displaying thumbnail PICT
Date Written: 2/21/92
Last reviewed: 1/5/93

How can I display the thumbnail of the PICT instead of some generic icon when I create

QuickTime PICT files? This would really help with distinguishing files when someone wanted to create a movie and had a lot of these PICTs around.

—

You need to follow these four steps:

1. Get the thumbnail. You can use either the MakeThumbnailFromPicture or MakeThumbnailFromPictureFile routines as listed in the ImageCompression interface file. It will pass back the PicHandle for the thumbnail. To install into the Finder, you need some resources (ICN#, ics#, icl8, ics8, icl4, ics4).

2. Make the thumbnail into a 'icsx' format to store it as a resource. (Please see MakeIcon on the Developer CD. It is not modified for pichandles so you may have to add a DrawPicture. Basically, you need to create a GWorld and create the appropriate 16- or 32-bit image.)

3. Add icons to resource. You can use the basic Resource Manager's WriteResource and AddResource calls to add the resource.

4. Set Finder bits: Stuff icon resources into the file itself with resource ID kCustomIconResource, and set the hasCustomIcon bit.

```
{ myCInfoPBRec.ioFlFndrInfo.fdFlags := BOR(myCInfoPBRec.ioFlFndrInfo.fdFlags, $0400) }.
```

QuickTime 1.0 FS6Patch elucidation
Date Written: 2/10/92
Last reviewed: 1/11/93

The FS6Patch description in the QuickTime 1.0 CD Read Me file says, "There is a known bug with HFS on System 6." What specifically is this bug? According to the Read Me file, "The exact situation under which the problem occurs is somewhat rare." What is the exact situation? "If the patch is not necessary, it will not install itself." What criteria is used to determine necessity?

—

With QuickTime 1.5 the patch has been incorporated into the QuickTime file instead of being a separate file, but the functionality, described below, is the same.

When closing or flushing a file that has multiple open paths to a given fork, it is possible that the blocks marked as available for the other paths can also be allocated to the other fork. If this happens before the other paths are closed, blocks will be mapped to more than one fork, potentially trashing at least one of them.

The condition that can trigger this condition is multiple writeable paths to the same fork. Once this is satisfied you would need to make some changes to the fork, close the path, use any of the other paths to modify the fork, access the other fork and modify data. If all these things happen you could see the damage to one of the forks (the second-to-last being accessed).

The code that loads the patch checks the system first; the range where the patch is used is 6.0.4 < system < 7.0. The version of the ROM is also used to decide when to install. If the ROM indicates the machine is later than the Macintosh IIfx, then a fix in ROM is assumed.

The last check is to see if the code being patched is already in RAM (the assumption is that being in RAM means fixed). If so, the patch doesn't get used.

QuickTime track and movie sound volume
Date Written: 2/3/92

Last reviewed: 1/11/93

What do the values of a movie's or track's volume represent? Is there no way to make a track louder?

—

Here's the scoop: The volume is described as a small fract 8:8 and its values go from -1 to 1 with negative values as placeholders. The maximum volume you can get is 0x0100 with the minimum being 0 (or any negative value). The advantage of using negative volumes is that you can turn off sound while maintaining the level of volume. For example, -1 and 0 both equate to no volume, but the -1 implies that 1 should be the volume when sound is turned back on, whereas the 0 does not.

The volume for a track is scaled to the movie's volume, and the movie's volume is scaled to the value the user specifies for the speaker volume using the Sound control panel. This means that the movie volume represents the maximum loudness of any track in the movie.

SetMovieRate and controlling movie playback rate
Date Written: 1/27/92
Last reviewed: 1/11/93

QuickTime is a joy! But I've run aground with SetMovieRate. I am trying to change the rate at which a movie plays back, but if I call SetMovieRate the movie starts playing immediately, the controller goes wild, and the next time I hit the play button it ignores the previous rate. How can I control my playback rate?

——

SetMovieRate takes effect immediately that is why your movies start playing as soon as you make the call with a rate not zero. Also calling the movie toolbox directly to change the state of a movie associated with a controller is a sure way to confuse the controller, causing the controller, for example, to show the play button when the movie is playing.

QuickTime 1.5 introduced a change in the movie controller which makes it use the current preferred movie rate when the user hits the play button. To change the normal playing rate you'll need to call SetMoviePreferredRate and let the controller do the rest.

Note that with QuickTime 1.0 the movie controller plays movies at rate of one (0x00010000) whenever the user hits the play button, regardless of the current preferred rate.

For details on filter procedures, controller actions, and the movie controller in general, see the "QuickTime Components" chapter of the *QuickTime Developer's Guide.*

Clipping QuickTime movie posters

Date Written: 12/10/91
Last reviewed: 1/11/93

Our application uses the movie poster as a still frame in a cell, similar to using a PICT. If a user sizes the cell width so that it's narrower than the poster, even though we clip the drawing to the cell size, QuickTime posters draw their full width, writing over whatever is in the way.

Pictures clip through DrawPicture; why doesn't ShowMoviePoster stay within the clipping region?

—

ShowMoviePoster, as well as the movie and preview showing calls, uses the movie clipping characteristics rather than the destination port's clipping region. You must set the movie's clipping region to obtain the results you want. An easier way to do this is to get the picture for the poster by calling GetMoviePosterPict, and then simply use DrawPicture to display the poster. Because this is just a picture, the clipping region of the port is honored. This way you don't need different code for movies and pictures.

PutMovieIntoHandle and data forks
Date Written: 12/10/91
Last reviewed: 1/11/93

I save PICTs to my document's data fork by writing the contents of the PicHandle. To save movies, do I convert the movie to a handle, and then save that as I would with PICTs? I just want the file references, not the data itself.

—

To save movies that are suitable for storage in a file, use PutMovieIntoHandle. The result of this call can be saved in the data fork of your files, and then you can call NewMovieFromHandle to reconstruct the movie for playback or editing.

You should also read the documentation regarding the Movie Toolbox FlattenMovie procedure, which creates a file that contains the 'moov' resource and the data all in the data fork. The advantage here is that the movie file you create using FlattenMovie can be read by any other QuickTime-capable application.

QuickTime and sound channel deallocation
Date Written: 12/2/91
Last reviewed: 1/11/93

Our QuickTime application gets a Sound Manager error -201 after playing movies in succession, apparently because sound channels used in the previous movies have not been reclaimed. How does QuickTime decide to deallocate sound channels? It doesn't seem to happen in my "while (!IsMovieDone(theMovie) && !Button())" play loop.

—

Sound channels are released by active movies when they notice that some other movie needs them. This is currently done only at MoviesTask time. Before entering your loop to play a single movie, you can do one or both of the following:

• Preroll the movie you're about to play and check the error. If preroll returns -201, call MoviesTask(0,0) to give the other active movies a chance to give up their sound channels. A subsequent preroll of theMovie should return noErr.
• Call SetMovieActive(otherMovies, FALSE). Deactivate the movies that you aren't playing to force them to give up their resources.

Standard controller with MCCut or MCClear
Date Written: 11/25/91
Last reviewed: 1/11/93

When I select all frames in QuickTime and then do an MCCut or MCClear, the standard controller gets larger and redraws itself at the top of the movie. Is this a situation I should be prepared to handle or a bug? Does the controller behave strangely when the selectionTime of a movie is -1 or when the duration of the movie is 0?

—

The behavior you're observing is to be expected if the controller is attached to the movie. In this case, the controller goes to wherever the bottom left corner of the movie box takes it. If the movie loses all its "visible" parts, the movie controller will jump to the top of the window. The only way to get around this is to detach the controller when the movie box is empty; this is also something to keep in mind for the cases when the movie contains only sound, since pure sound movies have no dimensions. You can find sample code showing how to do this on the *Developer CD Series* disc, in the SimpleInMovies example that accompanies the QuickTime article in *develop* Issue 7.

MCSetClip and clipping with the movie controller
Date Written: 11/15/91
Last reviewed: 1/11/93

I use SetMovieDisplayClipRgn to set my movie clip, but the movie doesn't obey my clipping. Does the movie controller component ignore this clipping?

—

You probably are directly modifying a movie that is attached to a controller without notifying the controller of the changes. The controller uses the display clip for its own purposes, such as for badges.

If you want to do clipping with the movie controller you must use MCSetClip. MCSetClip takes two regions. The first clips both the movie and the controller. The second clips just the movie, and is equivalent to the movie display clip. If both clips are set, the controller does the right thing and merges them as appropriate. If you don't want one or the other of the clips, set them to zero.

In general, if you are going to do something to a movie that is attached to a controller you must either do it through the controller, using the action calls, or you must call MCMovieChanged. Otherwise, the controller would need to constantly poll the movie to see if its state changed. Clearly this would be Evil and Slow.

QuickTime interfaces for Think Pascal

Date Written: 10/14/91
Last reviewed: 1/11/93

Has Apple created QuickTime headers/interfaces for Think Pascal? The MPW Pascal headers don’t seem to be compatible.

___

There are no Think Pascal interfaces, but your Pascal Package comes with a program called Pascal Source Converter, which converts MPW Pascal sources to Think. This should be all you need to be able to use the provided interfaces with your favorite development package.

How to get the first video frame
Date Written: 10/11/91
Last reviewed: 1/11/93

Stepping through QuickTime movie video frames in the order they appear in the movie is simple using GetMovieNextInterestingTime, except for getting the first frame. If I set the time to 0 and rate to 1, I get the second frame, not the first. In addition, the video may start later than at 0. How do you suggest finding this first frame of video?

—

To get the first frame under the conditions you describe, you have to pass the flag nextTimeEdgeOK = $2000 to GetMovieNextInterestingTime. What this flag does is make the call return the current interesting time instead of the next, if the current time is an interesting time. You need to do this because there's no way to go negative and then ask for the next interesting time.

Status of rotating matrix support
Date Written: 10/11/91
Last reviewed: 1/11/93

What's the status of RotateMatrix and its use with SetMovieMatrix and SetTrackMatrix?

—

RotateMatrix works fine. But rotating matrixes are not supported for movies or images. So, although RotateMatrix will give you the correct mathematical result, unless you are using the matrix to transform something else (as with TransformFixedPoints) it has little use.

Rotation is a very important direction that is sure to get more attention in the future.

QuickTime 1.5 supports user-defined media types
Date Written: 10/10/91
Last reviewed: 1/11/93

Can I add a media to a QuickTime movie that is not video or audio? If so, is there anything special I need to do to add text notes that can potentially accompany each frame in my "movie," which can follow the video frames if a user edits the movie in any way?

—

QuickTime version 1.0 allows for only video and sound media. There's no way to install your own type, even in a case so obvious as the one you mention. Adding other media types is a high QuickTime priority and is likely to make it in a future release, but currently there's no mechanism to do it.

QuickTime file audio retrieval
Date Written: 9/17/91
Last reviewed: 1/11/93

How can I retrieve audio from QuickTime files in 1-second chunks? I need a sound equivalent of GetMoviePict.

—

QuickTime 1.5 allows you to retrieve sound from a movie (or a single track from a movie) using the call ConvertMovieToFile. The developer QuickTime 1.5 CD contains a sample that shows how to do this; the path is "QuickTime 1.5: Programming Stuff: SampleCode: ImportExportMovie."

With QuickTime 1.0, you must write your own audio extraction routine, using GetMediaSample to collect the sound data by hand and the SoundDescription data structure to (1) find frequency, format, and other details, (2) help you figure out how many samples you need for a second, and (3) fill in the resource data fields.

See the QuickTime documentation for detailed information on ConvertMovieToFile. The best source of information on sound resources and such is the Sound Manager chapter in *Inside Macintosh* Volume VI.

QuickTime Movie Toolbox "globals" are stored in system heap
Date Written: 6/6/91
Last reviewed: 1/11/93

According to the QuickTime Movie Toolbox documentation, "The Movie Toolbox maintains a set of global variables for every application using it." How much global memory is required? Our application is shy on global data space.

—

The information maintained is not kept with the application's global variables. The handle created by the EnterMovies call is stored in the system heap, not in the application heap. You don't have to worry about how much space to allocate in your application. This initialization does not affect your A5 world either.

EnterMovies initializes everything, including setting up the necessary data space and creating a handle to it. When you're done, be sure to make a call to ExitMovies to clean up the QuickTime data.

If an application makes multiple calls to EnterMovies, a different set of "globals," or data area, is set up for each call. A call to ExitMovies should be made before exiting the area that made the call to EnterMovies. For example, an application that uses QuickTime will call EnterMovies and set up the QuickTime world. Then an external may be called upon that

wants to use QuickTime. This external would have to make a call to EnterMovies to set up another QuickTime world for its use. Before leaving, the external should call ExitMovies to clean up after itself. The application can then continue to use QuickTime with the original world it set up.